SMC Provider

CONTEC SMC Board

Version 1.0.3

User's Guide

October 25, 2012

[Remarks]

[Revision history]

Version	Date	Content
1.0.0.0	2011-7-27	First edition.
1.0.1.0	2011-9-27	Manual was corrected.
1.0.2.0	2012-5-29	Meta mode was added.
1.0.2	2012-7-17	Version rule of the document was changed.
1.0.3	2012-10-25	"ProviderCancel" and "ProviderClear" commands were added.

[Hardware]

Model	Version	Notes
SMC-4DL-PE		
SMC-4DF-PCI		

Contents

1. Introduction 4

2. Outline of Provider 5

2.1. Outline 5
2.2. Methods and properties 6
2.2.1. CaoWorkspace::AddController method 6
2.2.2. CaoController::Execute method 6
2.2.3. CaoController::AddVariable method 6
2.2.4. CaoController::GetVariableNames property 7
2.2.5. CaoController::AddExtension method 7
2.2.6. CaoController::GetExtensionNames property 8
2.2.7. CaoExtension::Execute method 8
2.2.8. CaoExtension::AddVariable method 8
2.2.9. CaoExtension::GetVariableNames property 8
2.2.10. CaoVariable::get_Value property 8
2.2.11. CaoVariable::put_Value property 8
2.3. List of commands 9
2.3.1. Controller class 9
2.3.2. Extension board class 9
2.4. Variable list 11
2.4.1. Controller class 11
2.4.2. Extension board class 11
2.5. Error code 19
2.6. CAO-SMC API reference table 19

3. Sample Program 22

1. Introduction

This document is a user's guide of the SMC provider which is used to access CONTEC SMC board.
Refer to CONTEC API-SMC (WDM) Help for details.

NOTE: SMC device driver of the SMC board needs to be installed to use the SMC provider.
Install the driver from CONTEC API-PAC (W32). After installing it, register the provider in the registry with reference to Table 2-1.

2. Outline of Provider

2.1. Outline

The SMC provider executes CONTEC API corresponding to CAO API at the time the CAO API is executed.
Refer to Table 2-10 for CAO API and corresponding CONTEC API.

Table 2-1 SMC provider

File name	CaoProvSMC.dll
ProgID	CaoProv.CONTEC.SMC
Registry registration[1]	regsvr32 CaoProvSMC.dll
Remove registry registration	regsvr32 /u CaoProvSMC.dll

[1] The SMC board driver must be installed to register the SMC provider.

2.2. Methods and properties

2.2.1. CaoWorkspace::AddController method

The SMC provider establishes (opens) connection to the SMC board when the Controller object is created.

Syntax AddController(<bstrCtrlName:BSTR>, <bstrProvName:BSTR>, <bstrPcName:BSTR >
[, <bstrOption:BSTR>])

bstrCtrlName : [in] Controller name
bstrProvName : [in] Provider name (Fixed to "CaoProv.CONTEC.SMC")
bstrPcName : [in] Provider execution machine name
bstrOption : [in] Option character string

The machine name can be an empty string.
Following is a list of option string items.

Table 2-2 Option character string of CaoWorkspace::AddController

Option	Meaning
DeviceName=<Device name>	Device name of the board to be connected Note: Specify the device supporting the SMC board ID.*1

*1: Refer to API-SMC (WDM) Help for details.

2.2.2. CaoController::Execute method

Refer to Table 2-7 for details of available commands and parameters.

Syntax Execute(< bstrCommand:BSTR > [,<vntParam:VARIANT>[,< pVal:VARIANT>]])

bstrCommand : [in] Command name
vntParam : [in] Parameter
pVal : [out] Acquired data

2.2.3. CaoController::AddVariable method

Creates a variable object used to access the SMC board. Only the variables given in 2.4.1 can be used. If a variable other than those is specified, this method will return an error.

Syntax AddVariable(<bstrName:BSTR > [, <bstrOption:BSTR>])

bstrName : [in] Arbitrary name
bstrOption : [in] Option character string (not used)

2.2.4. CaoController::GetVariableNames property

Acquires the variable name list in 2.4.1.

2.2.5. CaoController::AddExtension method

Creates CaoExtension that operates the SMC extension board.

Syntax AddExtension (<bstrName:BSTR > [,<bstrOption:BSTR>])

bstrName : [in] Extension board name

bstrOption : [in] Option character string

The table below shows available "extension board name".

Table 2-3 Extension board name list

Extension board name	Data type	Explanation
Axis?	VT_ BSTR	Specify the control target motor axis number (1 or higher)*1 of the SMC board at "?". Specify the logical number after the variable name. Example: "Axis1"

*1: The number of axes differs by the SMC board model. Refer to API-SMC (WDM) Help for details.

The table below shows available "option character string".

Table 2-4 Option character string of CaoController: AddExtension

Option	Meaning
MotorOut[=<Motor output>]	Allocate general output signal* for motor output destination. 0: Not allocated (default) 1: General output signal 1 (OUT1) 2: General output signal 2 (OUT2) 3: General output signal 3 (OUT3)

* Refer to API-SMC (WDM) Help for details.

2.2.6. CaoController::GetExtensionNames property

Acquires the extension board name list in 2.2.5.

2.2.7. CaoExtension::Execute method

Syntax Execute(< bstrCommand:BSTR > [,<vntParam:VARIANT>[,< pVal:VARIANT>]])

bstrCommand : [in] Command name

vntParam : [in] Parameter

pVal : [out] Acquired data

Refer to Table 2-6 for available "command name".

2.2.8. CaoExtension::AddVariable method

Creates a variable object used to access the specified axis of SMC board. Only the variables given in 2.4.2 can be used. If a variable other than those is specified, this method will return an error.

Syntax AddVariable(<bstrName:BSTR > [, <bstrOption:BSTR>])

bstrName : [in] Arbitrary name

bstrOption : [in] Option character string (not used)

2.2.9. CaoExtension::GetVariableNames property

Acquires the variable name list in 2.4.2.

2.2.10. CaoVariable::get_Value property

Acquires information corresponding to a variable. For the implementation status and acquired data of each variable, refer to 2.4.2.

2.2.11. CaoVariable::put_Value property

Configures information corresponding to a variable. For the implementation status and setting data of each variable, refer to 2.4.2.

2.3. List of commands

2.3.1. Controller class

Table 2-5 CaoController::Execute command list

Command name	Data type	Parameter	Acquired data	Explanation
ProviderCancel	—	—	—	Set cancel state [For the case all objects (Axis1...) are set to MotorOut = 0] Nothing is executed. A normal execution result (S_OK) is returned. [For the case other than MotorOut = 0] The "STOP" command is executed after the motor output signal of the object (Axis1...) is forced to turn OFF. An error execution result "E_PROV_CANCEL" is returned.
ProviderClear	—	—	—	Clear cancel state Nothing is executed. A normal execution result (S_OK) is returned.

2.3.2. Extension board class

Table 2-6 List of commands of CaoExtension::Execute

Command name	Data type	Parameter[*1]	Acquired data	Explanation
STOP	—	—	—	Stop
DSTP	—	—	—	Decelerated and stopped
ALMCLR	—	—	—	Output alarm clear signal pulse
ERCOUT	VT_I2	0: Output ERC signal. 1: Reset ERC signal.	—	Output error counter clear (ERC) signal
ORG	VT_I2	Operation direction (0: CW, 1: CCW)	—	Return to origin

MOVP	VT_ARRAY|VT_I4	<Element 1:Coordinate> Set the coordinate type of the position. 0: Absolute coordinate, 1: Relative coordinate <Element 2:StopPosition> Set the stop position. - Settable range - Absolute coordinate:-134,217,728 to +134,217,728 Relative coordinate: -134,217,728 to +134,217,72(other than 0)	—	Move to specified position
MOVJ	VT_I2	Operation direction (0: CW, 1: CCW)	—	JOG move
MCHG	VT_ARRAY| VT_I4	Motor operation change while in operation <Element 1:ChangeType> Set the motor operation change type. 0: Change to starting speed (FL speed) immediately 1: Change to target speed (FH speed) immediately 2: Change to starting speed (FL speed) after deceleration 3: Change to target speed (FH speed) after acceleration 4: Change operation speed and acceleration/deceleration 5: Change motor stop position 6: Change motor stop position by PCS signal <Element 2:Coordinate > Required only when Element 1 is 5 Set the coordinate type of the position. 0: Absolute coordinate, 1: Relative coordinate <Element 3:StopPosition > Required only when Element 1 is 5 Set the stop position. - Settable range - Absolute coordinate: -134,217,728 to +134,217,728 Relative coordinate: -134,217,728 to +134,217,72(other than 0)	—	Speed or stop position change during operation

*1: Refer to API-SMC (WDM) Help for details.

2.4. Variable list

2.4.1. Controller class

Table 2-7 Controller class system variable list

Variable name	Data type	Explanation	Attribute	
			get	put
@ERROR	VT_I4	Read the last error code*1 of SMC driver function.	√	—

*1: Refer to API-SMC (WDM) Help for details.

2.4.2. Extension board class

Table 2-8 Extension board class system variable list

Variable name	Data type	Explanation	Attribute	
			get	put
@INI_PLS	VT_ARRAY| VT_I2	Set/acquire an initial setting parameter related to pulse output.*1 <Element 1:PulseMode> Set the pulse output mode. 0: Common pulse type OUT: Negative logic, DIR+: High, DIR-: Low 1: Common pulse type OUT: Positive logic, DIR+: High, DIR-: Low 2: Common pulse type OUT: Negative logic, DIR+: Low, DIR-: High 3: Common pulse type OUT: Positive logic, DIR+: Low, DIR-: High 4: 2-pulse type: Negative logic (default) 5: 2-pulse type: Positive logic 6: 90-degree phase difference mode OUT: Advanced signal, DIR: Delayed signal 7: 90-degree phase difference mode OUT: Delayed signal, DIR: Advanced signal <Element 2:DirTimer>] Insert weight in changing direction. 0: OFF, 1: ON (default) Note: Available for common pulse type only [<Element 3:Duty>] Set duty ratio. 0: Vary by pulse output speed 1: Duty ratio fixed at 50% Default: DL series "1", DF series "0" Note: Specifying [omissible element] or "-1" makes the default effective.	√	√

@INI_CNT	VT_ARRAY| VT_I2	Set/acquire an initial setting parameter of counter operation.*1 <Element 1:ClearCntLtc> Clear counter when LTC signal changes from OFF to ON. Set the counter type. 0: Counter not cleared (default) 1: Clear output pulse counter 2: Clear encoder counter 3: Clear output pulse counter and encoder counter [<Element 2:LtcMode>] Set the counter type latched at LTC signal input. 0: Latch function not used (default) 1: Latch output pulse counter 2: Latch encoder counter 3: Latch output pulse counter and encoder counter [<Element 3:ClearCntClr>] Clear counter when CLR signal changes from OFF to ON. Set the counter type. 0: Counter not cleared (default) 1: Clear output pulse counter 2: Clear encoder counter 3: Clear output pulse counter and encoder counter Note: Specifying [omissible element] or "-1" makes the default effective.	√	√
@INI_DIO	VT_ARRAY| VT_I2	Set/acquire an initial setting parameter related to control input/output signal.*1 <Element 1:IoLog> Set logic of control input/output signal. [0 | 0 | 0 | 0 | 0 | OUT3 | OUT2 | OUT1| LIM | IN7 | IN6 | IN5 | IN4 | IN3 | IN2 | IN1] Setting range: 0 to 7FF (Hex) Setting value of each bit: 0 Negative logic, 1 Positive logic Default: 0 (negative logic for all) [<Element 2:InType>] Set the control input signal type (general input/ALM, INP, SD, LTC, PCS, CLR). [0 | 0 | IN6/CLR | IN5/PCS | IN4/LTC | IN3/SD | IN2/INP | IN1/ALM] Setting range: 0 to 3F (Hex) Default: 1 (IN1 used for alarm (ALM) signal input) [<Element 3:Out1>] [<Element 4:Out2>] [<Element 5:Out3>] Set the control output signal type for OUT1 to 3. 0: General output (default) 1: Alarm clear signal 2: Error counter clear signal (ERC) 3: Count match signal of output pulse counter (CP1) 4: Count match signal of encoder counter (CP2) 5: Hold off signal Note: Specifying [omissible element] or "-1" makes the default effective.	√	√

@INI_ENC	VT_ARRAY| VT_I2	Set/acquire an initial setting parameter related to encoder.*1 <Element 1:EncType> Set the encoder input type. 0: A/B 1 multiplication (default), 1: A/B 2 multiplication, 2: A/B 4 multiplication, 3: U/D, 4: Not used [<Element 2:ErcTime>] Set the width of error counter clear signal. 0: 12[µsec] (default), 1: 102[µsec], 2: 408[µsec], 3: 1.6[msec], 4: 13[msec], 5: 52[msec], 6: 104[msec], 7: Level output [<Element 3:ErcOffTimer>] Set OFF timer for error counter clear signal . 0: 0[µsec] (default), 1: 12[µsec], 2: 1.6[msec], 3: 104[msec] [<Element 4:AlmTime>] Set the width of alarm clear signal. 0: 12[µsec] (default), 1: 102[µsec], 2: 408[µsec], 3: 1.6[msec], 4: 13[msec], 5: 52[msec], 6: 104[msec] [<Element 5:ErcMode>] Specify ERC signal automatic output setting. [0 | 0 | 0 | 0 | 0 | 0 | bit1 | bit0] Setting range: 0 to 3 (Hex) bit0 0: ERC signal not output when LIM or ALM signal input causes a stop. 1: ERC signal output automatically when LIM or ALM signal input causes a stop. bit1 0: ERC signal not output when origin return is completed. 1: ERC signal output automatically when origin return is completed. Default: 0 Note: Specifying [omissible element] or "-1" makes the default effective.	√	√

@INI_ORG	VT_ARRAY| VT_I2	Set/acquire an initial setting parameter related to origin return.*1 <Element 1:OrgLog> Set origin input logic and edge. 0: Falling edge, negative logic (default) 1: Falling edge, positive logic 2: Rising edge, negative logic 3: Rising edge, positive logic [<Element 2:LimitTurn>] Set whether or not to have limit inversion during origin return. 0: Limit inversion not performed 1: Limit inversion performed (default) [<Element 3:OrgType>] Set whether or not to use Z phase. 0: Not used (ORG only) (default) 1: Used (ORG + Z phase) Note: Setting "0" disables ZCount setting. [<Element 4:EndDir>] Set origin entry direction for origin return (end direction of origin return). 0: Not specified (default) 1: Positive direction (CW) 2: Negative direction (CCW) [<Element 5:ZCount>] Set the number of Z phase for origin return. Setting range: 1 to 16 Default: 1 Note: Specifying "0" disables this setting. OrgType is changed automatically to "0" [Not used]. Note: Specifying [omissible element] or "-1" makes the default effective.	√	√
@INI_EXT	VT_ARRAY| VT_I2	Set/acquire other initial setting parameters.*1 <Element 1:SAccelType> Set whether or not to use S-curve acceleration/deceleration. 0: Not used (default), 1: Used [<Element 2:SdMode>] Set operation at SD signal input. 0: Decelerated to stop (default) 1: Deceleration only (constant speed operation performed with starting speed) 2: Decelerated and stopped + SD signal latched 3: Deceleration + SD signal latched [<Element 3:FilterType >] Set input filter characteristic. 0: Filter not inserted (default) 1:3.2[・ec], 2:25[・ec], 3:200[・ec], 4:1.6[msec] Note: Specifying [omissible element] or "-1" makes the default effective.	√	√

@MV_RDY	VT_ARRAY| VT_I2	Prepare for starting basic operation and acquire motor operation type/operation start direction.*1 <Element 1:MotionType> Set the motor operation type. 0: No operation, 1: PTP operation, 2: JOG operation, 3: Origin return operation, 6: Z phase counting operation <Element 2:StartDir> Acquire motor operation start direction. 0: Positive direction (CW), 1: Negative direction (CCW) Note: This variable is used for acquisition only. Setup will be performed automatically when a MOV command is executed.	√	—
@MV_CHGRDY	VT_I2	Acquire the motor operation change type.*1 - Motor operation change type - 0: Change to starting speed (FL speed) immediately 1: Change to target speed (FH speed) immediately 2: Change to starting speed (FL speed) after deceleration 3: Change to target speed (FH speed) after acceleration 4: Change operation speed and acceleration/deceleration 5: Change motor stop position 6 : Change motor stop position by PCS signal Note: This variable is used for acquisition only. Setup will be performed automatically when a MCHG command is executed.	√	—
@MV_STSPD	VT_R8	Set/acquire starting speed of pulse output.*1 Unit: PPS Default (100) Operation speed setting range: SMC-4/8DL series: 0.2929687 to 9829800 SMC-4/8DF series: 0.073242187 to 6553500 Note: Specifying "-1" makes the default effective.	√	√
@MV_TGSPD	VT_R8	Set/acquire target speed of pulse output.*1 Unit: PPS Default (1000) Operation speed setting range: SMC-4/8DL series: 0.2929687 to 9829800 SMC-4/8DF series: 0.073242187 to 6553500 Note: Specifying "-1" makes the default effective.	√	√
@MV_ACCTM	VT_R8	Set/acquire acceleration time.*1 Unit: ms Default (50) 0: Set to target speed immediately without taking acceleration process Note: Specifying "-1" makes the default effective.	√	√
@MV_DECTM	VT_R8	Set/acquire deceleration time.*1 Unit: ms Default (50) 0: Use the value set by @MV_ACCTM as deceleration time Note: Specifying "-1" makes the default effective.	√	√

@MV_RESPD	VT_R8	Set/acquire speed resolution.*1 Unit: PPS Default (1) Speed resolution set range: SMC-4/8DL series: -1, 0.2929687 to 600 SMC-4/8DF series: -1, 0.073242187 to 100 Note: Specifying "-1" makes the default effective.	√	√
@MV_SFSPD	VT_R8	Set/acquire S-curve zone.*1 Unit: PPS Default (400) 0: Perform S-curve operation without straight zone (maximum S-curve operation) Note: Specifying "-1" makes the default effective. This setting becomes effective when S-curve acceleration/deceleration is set to "1: Used" by @INI_EXT.	√	√
@MV_STPPOS	VT_ARRAY| VT_I4	Acquire motor stop position (total output pulse). <Element 1:StopPositionAbs > Stop position of absolute coordinate Absolute coordinate range: -134,217,728 to +134,217,727 <Element 2:StopPositionRel> Stop position of relative coordinate Relative coordinate range: -134,217,728 to +134,217,727 (other than 0) Note: This variable is used for acquisition only. Setup will be performed automatically when "MOVP" or "MCHG" command is executed.	√	—
@MV_ZCNT	VT_ARRAY| VT_I2	Set/acquire Z phase counting operation.*1 <Element 1:ZMoveCount> Set Z phase count number used in Z phase counting operation. Default (1) Setting range: 1 to 16 [<Element 2:ZLog>] Set input signal logic for Z phase signal. 0: Falling edge (default), 1: Rising edge Note: Specifying [omissible element] or "-1" makes the default effective. This setting cannot be used on SMC-4/8DL series.	√	√
@CNT_PLS	VT_I4	Acquire/set the number of feedback output pulses*1. Settable range: -134,217,728 to +134,217,727	√	√
@CNT_ENC	VT_I4	Acquire/set encoder count value*1. Settable range: -134,217,728 to +134,217,727	√	√
@CNT_ENZ	VT_I2	Acquire Z phase count in Z phase counting operation executed at origin return, or Z phase count in Z phase counting operation*1.	√	—
@LTC_PLS	VT_I4	Acquire the number of feedback output pulses*1 latched by LTC signal input.	√	—
@LTC_ENC	VT_I4	Acquire encoder counter value*1 latched by LTC signal input.	√	—

@STS_PLS	VT_I2	Acquire pulse output status*1. 0: Pulse output stopped 1: Constant speed operation with starting speed (FL speed) in process 2: Constant speed operation with target speed (FH speed) in process 3: Waiting for synchronous start 4: Waiting for ERC timer to end 5: Waiting for direction change timer to end 6: Acceleration in process 7: Deceleration in process 8: Waiting for INP input	√	—
@STS_MOV	VT_I2	Acquire motor operation status*1. 0: Stopped 1: PTP operation in process 2: JOG operation in process 3: Origin return in process 4: Banking in process (single) 5: Banking in process (loop)	√	—
@STS_STP	VT_I2	Acquire cause of motor stop*1. 0: In operation 1: Stop command 2: Deceleration stop command 3: Stop of another axis 4: Alarm/emergency stop signal 5: Positive direction limit stop signal 6: Negative direction limit stop signal 7: Deceleration stop signal 255: End of operation function	√	—
@STS_LMT	VT_I2	Acquire limit status*1. bit7 [0 | 0 | 0 | SD | ORG | -LIM | +LIM | ALM] bit0 ALM Alarm/emergency stop limit +LIM Positive direction limit -LIM Negative direction limit ORG Origin limit SD Deceleration stop limit Setting value: 0 Disabled, 1 Enabled	√	—
@STS_SPD	VT_R8	Acquire pulse output speed during operation*1.	√	—

@LMT_MASK	- For put - VT_ARRAY| VT_I2 - For get - VT_I2	Set/acquire use of limit signal.*1 <Element 1:LimitMask> … used for both put and get Set/acquire use of limit signal. [0 | 0 | 0 | 0 | 0 | 0 | ALM | SD] - Meaning of signal - SD Deceleration stop limit ALM Alarm/emergency stop limit - Setting/acquired value - 0: Limit enabled, 1: Limit disabled <Element 2:LimitMaskEnable> … used for put only [0 | 0 | 0 | 0 | 0 | 0 | ALM | SD] - Meaning of signal - SD Deceleration stop limit ALM Alarm/emergency stop limit - Setting value- 0 No change, 1 Change	√	√
@HOLD_OFF	VT_I2	Set/acquire hold off signal.*1 0 Axis hold 1 Axis hold released Note: One of @INI_DIO Out1 to 3 must be set to "Hold off signal".	√	√
@ALM_CODE	VT_I2	Acquire alarm code.*1 Acquired value indicates the status of IN5 to IN7. If alarm code output is available from the driver unit, connect the driver unit to IN5 to IN7 on the board. Refer to the motor driver unit manual for details of alarm codes.	√	—
@SIG_DO	- For put - VT_ARRAY| VT_I2 - For get - VT_I2	Acquire/set general output signal data.*1 <Element 1:OutData > … used for both put and get Set general output signal data. [0 | 0 | 0 | 0 | 0 | OUT3 | OUT2 | OUT1] 1: OUT 1 bit, 2: OUT 2 bit, 4: OUT 3 bit <Element 2:OutDataEnable > … used for put only Set the bit to change data for general output signal. Only the bit that specifies general output for OUT1 to 3 signal types by @INI_DIO can be set to 1. [0 | 0 | 0 | 0 | 0 | OUT3 | OUT2 | OUT1] 0: Disabled, 1: Enabled	√	√
@SIG_DI	VT_I2	Acquire general input signal data.*1 [0 | IN7 | IN6 | IN5 | IN4 | IN3 | IN2 | IN1] 0: Signal OFF, 1: Signal ON	√	—

*1: Refer to API-SMC (WDM) Help for details.

2.5. Error code

The SMC provider has the following two types of unique error code.

1) Error returned from SMC API

An error number returned from SMC API is returned after masking it with "0x80100***".

Example: SMC API error 0xFFF -> SMC API error 0x80100FFF

Refer to CONTEC API-SMC (WDM) Help for details of SMC API.

2) Error returned from SMC provider itself

Table 2-9 SMC provider unique error code list

Error name	Error code	Explanation
E_PROV_CANCEL	0x80180001	Operation stop due to provider cancel

3) ORiN2 common errors

Refer to the error code section in "ORiN2 Programming Guide".

2.6. CAO-SMC API reference table

The SMC provider offers two ways to execute commands using CaoExtension::Execute and CaoVariable. The CaoExtension::Execute method executes API function that performs operation.

CaoVariable executes API function that sets/acquires a value.

Table 2-10 Controller class, extension board class, variable class and corresponding SMC API

CAO API		SMC API*
Class::method	Parameter/ command/ variable	
CaoWorkspace::AddController()	DeviceName	SmcWInit()
CaoWorkspaces::Remove()	-	SmcWExit()
CaoController::AddExtension ()	Axis?	All functions of SmcWGet group SmcWSetInitParam()
CaoExtension::Execute()	STOP	SmcWMotionStop()
	DSTP	SmcWMotionDecStop()
	ALMCLR	SmcWSetAlarmClear()
	ERCOUT	SmcWSetErcOut()
	ORG	SmcWSetReady() SmcWMotionStart()
	MOVP	SmcWSetStopPosition() SmcWSetReady() SmcWMotionStart()
	MOVJ	SmcWSetReady() SmcWMotionStart()

	MCHG	SmcWSetMotionChangeReady() SmcWMotionChange()
CaoVariable::get_Value()	@INI_PLS	SmcWGetPulseType() SmcWGetPulseDuty()
	@INI_CNT	SmcWGetCounterMode()
	@INI_DIO	SmcWGetCtrlInOutLog() SmcWGetCtrlTypeIn() SmcWGetCtrlTypeOut()
	@INI_ENC	SmcWGetEncType() SmcWGetErcAlmClearTime() SmcWGetErcMode()
	@INI_ORG	SmcWGetOrgLog() SmcWGetOrgMode()
	@INI_EXT	SmcWGetSAccelType() SmcWGetSDMode() SmcWGetInFilterType()
	@MV_RDY	SmcWGetReady()
	@MV_CHGRDY	SmcWGetMotionChangeReady()
	@MV_STSPD	SmcWGetStartSpeed()
	@MV_TGSPD	SmcWGetTargetSpeed()
	@MV_ACCTM	SmcWGetAccelTime()
	@MV_DECTM	SmcWGetDecelTime()
	@MV_RESPD	SmcWGetResolveSpeed()
	@MV_SFSPD	SmcWGetSSpeed()
	@MV_STPPOS	SmcWGetStopPosition()
	@MV_ZCNT	SmcWGetZCountMotion()
	@CNT_PLS	SmcWGetOutPulse()
	@CNT_ENC	SmcWGetCountPulse()
	@CNT_ENZ	SmcWGetZCount()
	@LTC_PLS	SmcWGetLatchOutPulse()
	@LTC_ENC	SmcWGetLatchCountPulse()
	@STS_PLS	SmcWGetPulseStatus()
	@STS_MOV	SmcWGetMoveStatus()
	@STS_STP	SmcWGetStopStatus()
	@STS_LMT	SmcWGetLimitStatus()
	@STS_SPD	SmcWGetMoveSpeed()
	@LMT_MASK	SmcWGetLimitMask()
	@HOLD_OFF	SmcWGetHoldOff()
	@ALM_CODE	SmcWGetAlarmCode()
	@SIG_DO	SmcWGetDigitalOut()
	@SIG_DI	SmcWGetDigitalIn()
	@ERROR	—
CaoVariable::put_Value()	@INI_PLS	SmcWSetPulseType() SmcWSetPulseDuty()
	@INI_CNT	SmcWSetCounterMode()
	@INI_DIO	SmcWSetCtrlInOutLog() SmcWSetCtrlTypeIn() SmcWSetCtrlTypeOut()
	@INI_ENC	SmcWSetEncType() SmcWSetErcAlmClearTime() SmcWSetErcMode()

	@INI_ORG	SmcWSetOrgLog() SmcWSetOrgMode()
	@INI_EXT	SmcWSetSAccelType() SmcWSetSDMode() SmcWSeInFilterType()
	@MV_RDY	－
	@MV_CHGRDY	－
	@MV_STSPD	SmcWSetStartSpeed()
	@MV_TGSPD	SmcWSetTargetSpeed()
	@MV_ACCTM	SmcWSetAccelTime()
	@MV_DECTM	SmcWSetDecelTime()
	@MV_RESPD	SmcWSetResolveSpeed()
	@MV_SFSPD	SmcWSetSSpeed()
	@MV_STPPOS	SmcWSetStopPosition()
	@MV_ZCNT	SmcWSetZCountMotion()
	@CNT_PLS	SmcWSetOutPulse()
	@CNT_ENC	SmcWSetCountPulse()
	@CNT_ENZ	－
	@LTC_PLS	－
	@LTC_ENC	－
	@STS_PLS	－
	@STS_MOV	－
	@STS_STP	－
	@STS_LMT	－
	@STS_SPD	－
	@LMT_MASK	SmcWSetLimitMask()
	@HOLD_OFF	SmcWSetHoldOff()
	@ALM_CODE	－
	@SIG_DO	SmcWSetDigitalOut
	@SIG_DI	－
	@ERROR	－

* Refer to CONTEC API-SMC (WDM) Help for details of SMC API.

3. Sample Program

The following shows the code to execute origin return of the SMC board Axis No. 1 and move to the point No. 1 after checking that the origin return is completed.

List 3-1 Sample.frm

```
Dim Eng As CaoEngine
Dim Ctrl As CaoController
Dim ExtAxis As CaoExtension
Dim VarIniPls As CaoVariable
Dim VarMov As CaoVariable
Dim VarSpd As CaoVariable

Private Sub Form_Load()

        ' Create CAO engine
        Set Eng = New CaoEngine

        ' Connect to SMC
        Set Ctrl = Eng.Workspaces(0).AddController("Sample",, _
                                        "CaoProv.CONTEC.SMC", "", "DeviceName=SMC000")

        ' Create extension board class of Axis No. 1
        Set ExtAxis = Ctrl.AddExtension("Axis1")

        ' Create variables
        Set VarIniPls = ExtAxis.AddVariable("@INI_PLS")
                                        'Create system variable related to pulse output setting
        Set VarMov = ExtAxis.AddVariable("@STS_MOV")
                                        ' Create system variable for operation status check
        Set VarSpd = ExtAxis.AddVariable("@MV_TGSPD")
                                        ' Create system variable for target speed setting

        ' Initial setting related to Motor (driver)
        VarIniPls = Array(5)

        ' Origin return performed in CCW direction
        ExtAxis.Execute "ORG", 1

End Sub

Private Sub Command1_Click()

        ' Wait until origin return is completed
        Do While ( VarMov <> 0 )
            DoEvents
        Loop

        VarSpd = 500#               'Set target speed to 500PPS
        ExtAxis.Execute "MOVP", Array(1, 5000)
                                        'Move by 5000 pulses in + (CW) direction on relative
                                        ' coordinate

        'Wait during operation
        Do While ( VarMov <> 0 )
            DoEvents
        Loop

End Sub
```